Relación de Ejercicios de Arrays (4)

En esta relación se requiere modificar el tamaño de un array. Como en Java no se puede, lo que haremos será crear un array nuevo más grande o más pequeño y copiar en él los datos del primer array.

31. Escribe una función "concatenaArraysPro" a la que le pasas dos arrays de enteros y te devuelve un array cuyo tamaño es la suma del tamaño de ambos y que contiene todos los elementos del primero y a continuación los del segundo. Esta función devolverá un array de enteros (*int[]*) el cuál se creará dentro de la propia función.

    Ej. de uso:
    ```
    int[] a = {1,2,3};
    int[] b = {4,5,6,7,2};
    int[] c;
    c = concatenaArraysPro(a, b);
    ```

32. Escribe una función "copiaArrayPro" a la que le pasas un array y te devuelve un array del mismo tamaño y con los mismos datos.

    Ej. de uso:
    ```
    int[] a = {1,2,3};
    int[] b;
    b = copiaArrayPro(a);
    ```

33. Escribe una función "insertaEnArray" a la que le pasas tres parámetros: un array de enteros, un valor entero y una posición. La función insertará el valor en la posición indicada, desplazando el resto de valores para hacerle hueco. La función nos devolverá un array con el resultado. Ej.: Si tenemos el array [1,2,3,4,5] y queremos insertar el valor "26" en la posición "2", el resultado será: [1,2,26,3,4,5].

34. Escribe una función "borraDeArray" a la que le pasas dos parámetros: un array de enteros y una posición. La función eliminará el elemento colocado en la posición indicada. La función nos devolverá un array con el resultado. Ej.: Si tenemos el array [5,7,2,8,1] y queremos eliminar la posición 1, el resultado será: [5,2,8,1].

35. Escribe una función "elimina1ElementoArray" a la que le pasas dos parámetros: un array de enteros y un valor entero. La función eliminará del array el valor entero independientemente de la posición en la que se encuentre. Si el valor se repite, se eliminará sólo la primera vez que aparece el valor. La función nos devolverá un array con el resultado.

36. Escribe una función "eliminaElementosArray" a la que le pasas dos parámetros: un array de enteros y un valor entero. La función eliminará del array el valor entero independientemente de la posición en la que se encuentre. Si el valor se repite, se eliminarán todas las veces que se repite. La función nos devolverá un array con el resultado.

37. Escribe una función “insertaArrayEnArray” a la que le pasas tres parámetros: un array de enteros, una posición de ese array, y otro array de enteros. La función insertará en el primer array, a partir de la posición indicada, todo el contenido del segundo array. La función nos devolverá un array con el resultado. Ej.: [6, 2, 1, 3], posición: 2, insertar: [12, 13], resultado = [6, 2, 12, 13, 1, 3].

38. Escribe una función “subArray” a la que le pasas un array y dos posiciones. La función te devuelve otro array que contiene los datos comprendidos entre ambas posiciones (incluidas ambas dos). Ej.: [3, 6, 2, 8, 9], posiciones, 1 y 3, resultado = [6, 2, 8].

39. Escribe una función “recortaArray” a la que le pasas un array y dos posiciones. La función te devuelve otro array que contiene los todos los datos menos los comprendidos entre ambas posiciones (incluidas ambas dos). Ej.: [3, 6, 2, 8, 9], posiciones, 1 y 3, resultado = [3, 9].

40. Escribe la función “ordenaBurbuja” a la que le pasaremos un array de enteros y lo ordenará mediante la ordenación de la burbuja. La ordenación de la burbuja funciona de la siguiente forma:
    - Si el array tiene N elementos, se realizarán N-1 pasadas (por ejemplo, para un array de 5 elementos tendremos que realizar 4 pasadas para que quede ordenado).
    - En cada pasada, iremos recorriendo el array de principio a fin comprobando los números por parejas (por ejemplo: el primer y segundo número, el segundo y el tercero, etc.).
    - Si la pareja de números está en orden (o sea, el primero es menor que el segundo) la dejamos como está y si está desordenada, los intercambiamos.

    Ejemplo:

    Si el array original es [3, 9, 4, 6, 8, 1], la pasada sería así:

    [**3**, **9**, 4, 6, 8, 1] -> [3, **4**, **9**, 6, 8, 1] -> [3, 4, **6**, **9**, 8, 1] -> [3, 4, 6, **8**, **9**, 1] -> [3, 4, 6, 8, **1**, **9**]

    Podéis comprobar que el array no se ordena con una sola pasada, pero poco a poco va quedando más ordenado.